REJ09B0130-0090Z

16

M16C/26 Group

Usage Notes Reference Book

RENESAS 16-BIT SINGLE-CHIP MICROCOMPUTER

M16C FAMILY / M16C/20 SERIES

For the most current **Usage Notes Reference Book**, please visit our website.

Before using this material, please visit our website to confirm that this is the most current document available.

Rev. 0.90
Revision date: December. 28, 2003

Renesas Technology
www.renesas.com

Usage Notes Reference Book

Keep safety first in your circuit designs!

- Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
  The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
  Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation for further details on these materials or t he products contained therein.

Preface

The “Usage Notes Reference Book” is a compilation of usage notes from the Hardware Manual as well as technical news related to this product.

Table of contents

1. Usage Precaution

1.1 Precautions for Power Control

1. When exiting stop mode by hardware reset, set $\overline{\text{RESET}}$ pin to “L” until a main clock oscillation is stabilized.

2. Insert more than four NOP instructions after an WAIT instruction or a instruction to set the CM10 bit of CM1 register to “1”. When shifting to wait mode or stop mode, an instruction queue reads ahead to the next instruction to halt a program by an WAIT instruction and an instruction to set the CM10 bit to “1” (all clocks stopped). The next instruction may be executed before entering wait mode or stop mode, depending on a combination of instruction and an execution timing.

3. Wait until the $t_{d(M-L)}$ elapses or main clock oscillation stabilization time, whichever is longer, before switching the clock source for CPU clock to the main clock.
Similarly, wait until the sub clock oscillates stably before switching the clock source for CPU clock to the sub clock.

4. Suggestions to reduce power consumption

(a) Ports

The processor retains the state of each I/O port even when it goes to wait mode or to stop mode. A current flows in active I/O ports. A pass current flows in input ports that high-impedance state. When entering wait mode or stop mode, set non-used ports to input and stabilize the potential.

(b) A-D converter

When A-D conversion is not performed, set the VCUT bit of ADiCON1 register to “0” (no V_{REF} connection). When A-D conversion is performed, start the A-D conversion at least 1 μs or longer after setting the VCUT bit to “1” (V_{REF} connection).

(c) Stopping peripheral functions

Use the CM0 register CM02 bit to stop the unnecessary peripheral functions during wait mode.
However, because the peripheral function clock (f_{C32}) generated from the sub-clock does not stop, this measure is not conducive to reducing the power consumption of the chip. If low speed mode or low power dissipation mode is to be changed to wait mode, set the CM02 bit to “0” (do not peripheral function clock stopped when in wait mode), before changing wait mode.

(d) Switching the oscillation-driving capacity

Set the driving capacity to “LOW” when oscillation is stable.

(e) External clock

When using an external clock input for the CPU clock, set the CM0 register CM05 bit to “1” (stop). Setting the CM05 bit to “1” disables the X_{OUT} pin from functioning, which helps to reduce the amount of current drawn in the chip. (When using an external clock input, note that the clock remains fed into the chip regardless of how the CM05 bit is set.)

1.2 Precautions for Protect

Set the PRC2 bit to “1” (write enabled) and then write to any address, and the PRC2 bit will be cleared to “0” (write protected). The registers protected by the PRC2 bit should be changed in the next instruction after setting the PRC2 bit to “1”. Make sure no interrupts or DMA transfers will occur between the instruction in which the PRC2 bit is set to “1” and the next instruction.

1.3 Precautions for Interrupts

1.3.1 Reading address 00000$_{16}$

Do not read the address 00000$_{16}$ in a program. When a maskable interrupt request is accepted, the CPU reads interrupt information (interrupt number and interrupt request priority level) from the address 00000$_{16}$ during the interrupt sequence. At this time, the IR bit for the accepted interrupt is cleared to “0”. If the address 00000$_{16}$ is read in a program, the IR bit for the interrupt which has the highest priority among the enabled interrupts is cleared to “0”. This causes a problem that the interrupt is canceled, or an unexpected interrupt request is generated.

1.3.2 Setting the SP

Set any value in the SP(USP, ISP) before accepting an interrupt. The SP(USP, ISP) is cleared to ‘0000$_{16}$’ after reset. Therefore, if an interrupt is accepted before setting any value in the SP(USP, ISP), the program may go out of control.

Especially when using $\overline{\text{NMI}}$ interrupt, set a value in the ISP at the beginning of the program. For the first and only the first instruction after reset, all interrupts including $\overline{\text{NMI}}$ interrupt are disabled.

1.3.3 The $\overline{\text{NMI}}$ Interrupt

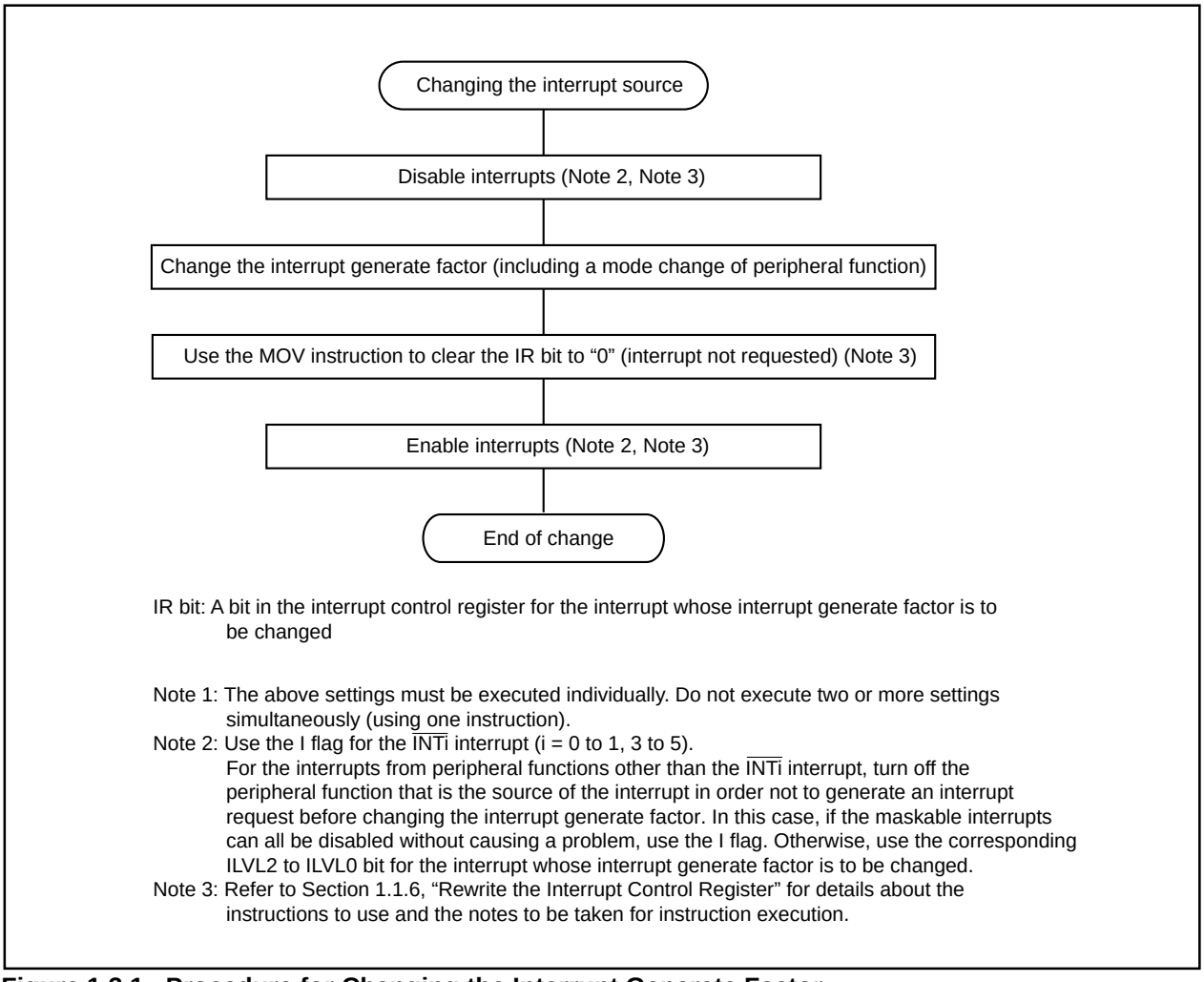
1. The $\overline{\text{NMI}}$ interrupt is invalid after reset. The $\overline{\text{NMI}}$ interrupt becomes effective by setting to “1” the PM24 bit of the PM2 register. Once enabled, it stays enabled until a reset is applied.
2. The input level of the $\overline{\text{NMI}}$ pin can be read by accessing the P8 register’s P8_5 bit. Note that the P8_5 bit can only be read when determining the pin level in $\overline{\text{NMI}}$ interrupt routine.
3. When selecting $\overline{\text{NMI}}$ function, stop mode cannot be entered into while input on the $\overline{\text{NMI}}$ pin is low. This is because while input on the $\overline{\text{NMI}}$ pin is low the CM1 register’s CM10 bit is fixed to “0”.
4. When selecting $\overline{\text{NMI}}$ function, do not go to wait mode while input on the $\overline{\text{NMI}}$ pin is low. This is because when input on the $\overline{\text{NMI}}$ pin goes low, the CPU stops but CPU clock remains active; therefore, the current consumption in the chip does not drop. In this case, normal condition is restored by an interrupt generated thereafter.
5. When selecting $\overline{\text{NMI}}$ function, the low and high level durations of the input signal to the $\overline{\text{NMI}}$ pin must each be 2 CPU clock cycles + 300 ns or more.

1.3.4 Changing the Interrupt Generate Factor

If the interrupt generate factor is changed, the IR bit in the interrupt control register for the changed interrupt may inadvertently be set to “1” (interrupt requested). If you changed the interrupt generate factor for an interrupt that needs to be used, be sure to clear the IR bit for that interrupt to “0” (interrupt not requested).

“Changing the interrupt generate factor” referred to here means any act of changing the source, polarity or timing of the interrupt assigned to each software interrupt number. Therefore, if a mode change of any peripheral function involves changing the generate factor, polarity or timing of an interrupt, be sure to clear the IR bit for that interrupt to “0” (interrupt not requested) after making such changes. Refer to the description of each peripheral function for details about the interrupts from peripheral functions.

Figure 1.3.1 shows the procedure for changing the interrupt generate factor.



IR bit: A bit in the interrupt control register for the interrupt whose interrupt generate factor is to be changed

Note 1: The above settings must be executed individually. Do not execute two or more settings simultaneously (using one instruction).
Note 2: Use the I flag for the $\overline{\text{INT}}$i interrupt (i = 0 to 1, 3 to 5).
For the interrupts from peripheral functions other than the $\overline{\text{INT}}$i interrupt, turn off the peripheral function that is the source of the interrupt in order not to generate an interrupt request before changing the interrupt generate factor. In this case, if the maskable interrupts can all be disabled without causing a problem, use the I flag. Otherwise, use the corresponding ILVL2 to ILVL0 bit for the interrupt whose interrupt generate factor is to be changed.
Note 3: Refer to Section 1.1.6, “Rewrite the Interrupt Control Register” for details about the instructions to use and the notes to be taken for instruction execution.

Figure 1.3.1. Procedure for Changing the Interrupt Generate Factor

1.3.5 $\overline{\text{INT}}$ Interrupt

1. Either an “L” level of at least $t_{W(INH)}$ or an “H” level of at least $t_{W(INL)}$ width is necessary for the signal input to pins $\overline{\text{INT}}_0$ through $\overline{\text{INT}}_1$, and $\overline{\text{INT}}_3$ through $\overline{\text{INT}}_5$ regardless of the CPU operation clock.
2. If the POL bit in the INT0IC to INT1IC and INT3IC to INT5IC registers or the IFSR7 to IFSR0 bits in the IFSR register are changed, the IR bit may inadvertently set to 1 (interrupt requested). Be sure to clear the IR bit to 0 (interrupt not requested) after changing any of those register bits.

1.3.6 Rewrite the Interrupt Control Register

(1) The interrupt control register for any interrupt should be modified in places where no requests for that interrupt may occur. Otherwise, disable the interrupt before rewriting the interrupt control register.

(2) To rewrite the interrupt control register for any interrupt after disabling that interrupt, be careful with the instruction to be used.

Changing any bit other than the IR bit

If while executing an instruction, a request for an interrupt controlled by the register being modified occurs, the IR bit in the register may not be set to “1” (interrupt requested), with the result that the interrupt request is ignored. If such a situation presents a problem, use the instructions shown below to modify the register.

Usable instructions: AND, OR, BCLR, BSET

Changing the IR bit

Depending on the instruction used, the IR bit may not always be cleared to “0” (interrupt not requested). Therefore, be sure to use the MOV instruction to clear the IR bit.

(3) When using the I flag to disable an interrupt, refer to the sample program fragments shown below as you set the I flag. (Refer to (2) for details about rewrite the interrupt control registers in the sample program fragments.)

Examples 1 through 3 show how to prevent the I flag from being set to “1” (interrupts enabled) before the interrupt control register is rewrited, owing to the effects of the internal bus and the instruction queue buffer.

Example 1:Using the NOP instruction to keep the program waiting until the interrupt control register is modified

```
INT_SWITCH1:
        FCLR    I               ; Disable interrupts.
        AND.B   #00h, 0055h     ; Set the TA0IC register to "00₁₆".
        NOP                     ;
        NOP
        FSET    I               ; Enable interrupts.
```

The number of NOP instruction is 2.

Example 2:Using the dummy read to keep the FSET instruction waiting

```
INT_SWITCH2:
        FCLR    I               ; Disable interrupts.
        AND.B   #00h, 0055h     ; Set the TA0IC register to "00₁₆".
        MOV.W   MEM, R0         ; Dummy read.
        FSET    I               ; Enable interrupts.
```

Example 3:Using the POPC instruction to changing the I flag

```
INT_SWITCH3:
        PUSHC   FLG
        FCLR    I               ; Disable interrupts.
        AND.B   #00h, 0055h     ; Set the TA0IC register to "00₁₆".
        POPC    FLG             ; Enable interrupts.
```

1.3.7 Watchdog Timer Interrupt

Initialize the watchdog timer after the watchdog timer interrupt occurs.

1.4 Precautions for DMAC

1.4.1 Write to DMAE Bit in DMiCON Register

When both of the conditions below are met, follow the steps below.

Conditions

- The DMAE bit is set to “1” again while it remains set (DMAi is in an active state).
- A DMA request may occur simultaneously when the DMAE bit is being written.

Step 1: Write “1” to the DMAE bit and DMAS bit in DMiCON register simultaneously(*1).
Step 2: Make sure that the DMAi is in an initial state(*2) in a program.
If the DMAi is not in an initial state, the above steps should be repeated.

Notes:

*1. The DMAS bit remains unchanged even if “1” is written. However, if “0” is written to this bit, it is set to “0” (DMA not requested). In order to prevent the DMAS bit from being modified to “0”, “1” should be written to the DMAS bit when “1” is written to the DMAE bit. In this way the state of the DMAS bit immediately before being written can be maintained.
Similarly, when writing to the DMAE bit with a read-modify-write instruction, “1” should be written to the DMAS bit in order to maintain a DMA request which is generated during execution.

*2. Read the TCRi register to verify whether the DMAi is in an initial state. If the read value is equal to a value which was written to the TCRi register before DMA transfer start, the DMAi is in an initial state. (If a DMA request occurs after writing to the DMAE bit, the value written to the TCRi register is “1”.) If the read value is a value in the middle of transfer, the DMAi is not in an initial state.

1.5 Precautions for Timers

1.5.1 Timer A

1.5.1.1 Timer A (Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register and the TAi register before setting the TAiS bit in the TABSR register to “1” (count starts).
   Always make sure the TAiMR register is modified while the TAiS bit remains “0” (count stops) regardless whether after reset or not.

2. While counting is in progress, the counter value can be read out at any time by reading the TAi register. However, if the counter is read at the same time it is reloaded, the value “$FFFF_{16}$” is read. Also, if the counter is read before it starts counting after a value is set in the TAi register while not counting, the set value is read.

3. If a low-level signal is applied to the $P8_5/\overline{NMI}/\overline{SD}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{SD}$ pin enabled), the $TA1_{OUT}$, $TA2_{OUT}$ and $TA4_{OUT}$ pins go to a high-impedance state.

1.5.1.2 Timer A (Event Counter Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to “1” (count starts).
   Always make sure the TAiMR register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains “0” (count stops) regardless whether after reset or not.

2. While counting is in progress, the counter value can be read out at any time by reading the TAi register. However, “FFFF$_{16}$” can be read in underflow, while reloading, and “0000$_{16}$” in overflow. When setting TAi register to a value during a counter stop, the setting value can be read before a counter starts counting. Also, if the counter is read before it starts counting after a value is set in the TAi register while not counting, the set value is read.

3. If a low-level signal is applied to the P8$_5$/$\overline{\text{NMI}}$/$\overline{\text{SD}}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin enabled), the TA1$_{\text{OUT}}$, TA2$_{\text{OUT}}$ and TA4$_{\text{OUT}}$ pins go to a high-impedance state.

1.5.1.3 Timer A (One-shot Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to “1” (count starts).
   Always make sure the TAiMR register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains “0” (count stops) regardless whether after reset or not.

2. When setting TAiS bit to “0” (count stop), the followings occur:
   • A counter stops counting and a content of reload register is reloaded.
   • TAi$_{OUT}$ pin outputs “L”.
   • After one cycle of the CPU clock, the IR bit of TAiIC register is set to “1” (interrupt request).

3. Output in one-shot timer mode synchronizes with a count source internally generated. When an external trigger has been selected, one-cycle delay of a count source as maximum occurs between a trigger input to TAi$_{IN}$ pin and output in one-shot timer mode.

4. The IR bit is set to “1” when timer operation mode is set with any of the following procedures:
   • Select one-shot timer mode after reset.
   • Change an operation mode from timer mode to one-shot timer mode.
   • Change an operation mode from event counter mode to one-shot timer mode.
   To use the timer Ai interrupt (the IR bit), set the IR bit to “0” after the changes listed above have been made.

5. When a trigger occurs, while counting, a counter reloads the reload register to continue counting after generating a re-trigger and counting down once. To generate a trigger while counting, generate a second trigger between occurring the previous trigger and operating longer than one cycle of a timer count source.

6. If a low-level signal is applied to the P8$_5$/$\overline{NMI}$/$\overline{SD}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{SD}$ pin enabled), the TA1$_{OUT}$, TA2$_{OUT}$ and TA4$_{OUT}$ pins go to a high-impedance state.

1.5.1.4 Timer A (Pulse Width Modulation Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to “1” (count starts).
   Always make sure the TAiMR register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains “0” (count stops) regardless whether after reset or not.

2. The IR bit is set to “1” when setting a timer operation mode with any of the following procedures:
   • Select the PWM mode after reset.
   • Change an operation mode from timer mode to PWM mode.
   • Change an operation mode from event counter mode to PWM mode.
   To use the timer Ai interrupt (interrupt request bit), set the IR bit to “0” by program after the above listed changes have been made.

3. When setting TAiS register to “0” (count stop) during PWM pulse output, the following action occurs:
   • Stop counting.
   • When TAiOUT pin is output “H”, output level is set to “L” and the IR bit is set to “1”.
   • When TAiOUT pin is output “L”, both output level and the IR bit remains unchanged.

4. If a low-level signal is applied to the P85/$\overline{\text{NMI}}$/$\overline{\text{SD}}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin enabled), the TA1OUT, TA2OUT and TA4OUT pins go to a high-impedance state.

1.5.2 Timer B

1.5.2.1 Timer B (Timer Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBiMR (i = 0 to 2) register and TBi register before setting the TBiS bit in the TABSR register to “1” (count starts).
   Always make sure the TBiMR register is modified while the TBiS bit remains “0” (count stops) regardless whether after reset or not.

2. A value of a counter, while counting, can be read in TBi register at any time. “FFFF$_{16}$” is read while reloading. Setting value is read between setting values in TBi register at count stop and starting a counter.

1.5.2.2 Timer B (Event Counter Mode)

1. The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBiMR (i = 0 to 2) register and TBi register before setting the TBiS bit in the TABSR register to “1” (count starts).
   Always make sure the TBiMR register is modified while the TBiS bit remains “0” (count stops) regardless whether after reset or not.

2. The counter value can be read out on-the-fly at any time by reading the TBi register. However, if this register is read at the same time the counter is reloaded, the read value is always “FFFF16.” If the TBi register is read after setting a value in it while not counting but before the counter starts counting, the read value is the one that has been set in the register.

1.5.2.3 Timer B (Pulse Period/pulse Width Measurement Mode)

1. The timer remains idle after reset. Set the mode, count source, etc. using the TBiMR (i = 0 to 2) register before setting the TBiS bit in the TABSR register to “1” (count starts).
   Always make sure the TBiMR register is modified while the TBiS bit remains “0” (count stops) regardless whether after reset or not. To clear the MR3 bit to “0” by writing to the TBiMR register while the TBiS bit = “1” (count starts), be sure to write the same value as previously written to the TM0D0, TM0D1, MR0, MR1, TCK0 and TCK1 bits and a 0 to the MR2 bit.

2. The IR bit of TBiIC register (i=0 to 2) goes to “1” (interrupt request), when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the MR3 bit of TBiMR register within the interrupt routine.

3. If the source of interrupt cannot be identified by the MR3 bit such as when the measurement pulse input and a timer overflow occur at the same time, use another timer to count the number of times timer B has overflowed.

4. To set the MR3 bit to “0” (no overflow), set TBiMR register with setting the TBiS bit to “1” and counting the next count source after setting the MR3 bit to “1” (overflow).

5. Use the IR bit of TBiIC register to detect only overflows. Use the MR3 bit only to determine the interrupt factor within the interrupt routine.

6. When a count is started and the first effective edge is input, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

7. A value of the counter is indeterminate at the beginning of a count. MR3 may be set to “1” and timer Bi interrupt request may be generated between a count start and an effective edge input.

8. For pulse width measurement, pulse widths are successively measured. Use program to check whether the measurement result is an “H” level width or an “L” level width.

1.6 Precautions for Serial I/O (Clock-synchronous Serial I/O)

1.6.1 Transmission/reception

1. With an external clock selected, and choosing the $\overline{RTS}$ function, the output level of the $\overline{RTSi}$ pin goes to “L” when the data-receivable status becomes ready, which informs the transmission side that the reception has become ready. The output level of the $\overline{RTSi}$ pin goes to “H” when reception starts. So if the $\overline{RTSi}$ pin is connected to the $\overline{CTSi}$ pin on the transmission side, the circuit can transmission and reception data with consistent timing. With the internal clock, the $\overline{RTS}$ function has no effect.

2. If a low-level signal is applied to the P8$_5$/$\overline{NMI}$/$\overline{SD}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{SD}$ pin enabled), the $\overline{RTS_2}$ and CLK$_2$ pins go to a high-impedance state.

1.6.2 Transmission

When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

- The TE bit of UiC1 register= "1" (transmission enabled)
- The TI bit of UiC1 register = "0" (data present in UiTB register)
- If $\overline{\text{CTS}}$ function is selected, input on the $\overline{\text{CTS}}$i pin = "L"

1.6.3 Reception

1. In operating the clock-synchronous serial I/O, operating a transmitter generates a shift clock. Fix settings for transmission even when using the device only for reception. Dummy data is output to the outside from the TxDi pin when receiving data.

2. When an internal clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 (transmission enabled) and write dummy data to the UiTB register, and the shift clock will thereby be generated. When an external clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 and write dummy data to the UiTB register, and the shift clock will be generated when the external clock is fed to the CLKi input pin.

3. When successively receiving data, if all bits of the next receive data are prepared in the UARTi receive register while the UiC1 register (i = 0 to 2)'s RE bit = "1" (data present in the UiRB register), an overrun error occurs and the UiRB register OER bit is set to "1" (overrun error occurred). In this case, because the content of the UiRB register is indeterminate, a corrective measure must be taken by programs on the transmit and receive sides so that the valid data before the overrun error occurred will be retransmitted. Note that when an overrun error occurred, the SiRIC register IR bit does not change state.

4. To receive data in succession, set dummy data in the lower-order byte of the UiTB register every time reception is made.

5. When an external clock is selected, the conditions must be met while if the CKPOL bit = "0", the external clock is in the high state; if the CKPOL bit = "1", the external clock is in the low state.
   - The RE bit of UiC1 register= "1" (reception enabled)
   - The TE bit of UiC1 register= "1" (transmission enabled)
   - The TI bit of UiC1 register= "0" (data present in the UiTB register)

1.7 Precautions for Serial I/O (UART Mode)

1.7.1 Special Mode 2

If a low-level signal is applied to the P8$_5$/$\overline{\text{NMI}}$/$\overline{\text{SD}}$ pin when the TB2SC register IVPCR1 bit = 1 (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin enabled), the $\overline{\text{RTS}}_2$ and CLK$_2$ pins go to a high-impedance state.

1.7.2 Special Mode 4 (SIM Mode)

A transmit interrupt request is generated by setting the U2C1 register U2IRS bit to “1” (transmission complete) and U2ERE bit to “1” (error signal output) after reset. Therefore, when using SIM mode, be sure to clear the IR bit to “0” (no interrupt request) after setting these bits.

1.8 Precautions for A-D Converter

1. Set ADCON0 (except bit 6), ADCON1 and ADCON2 registers when A-D conversion is stopped (before a trigger occurs).

2. When the VCUT bit of ADCON1 register is changed from “0” (Vref not connected) to “1” (Vref connected), start A-D conversion after passing 1 µs or longer.

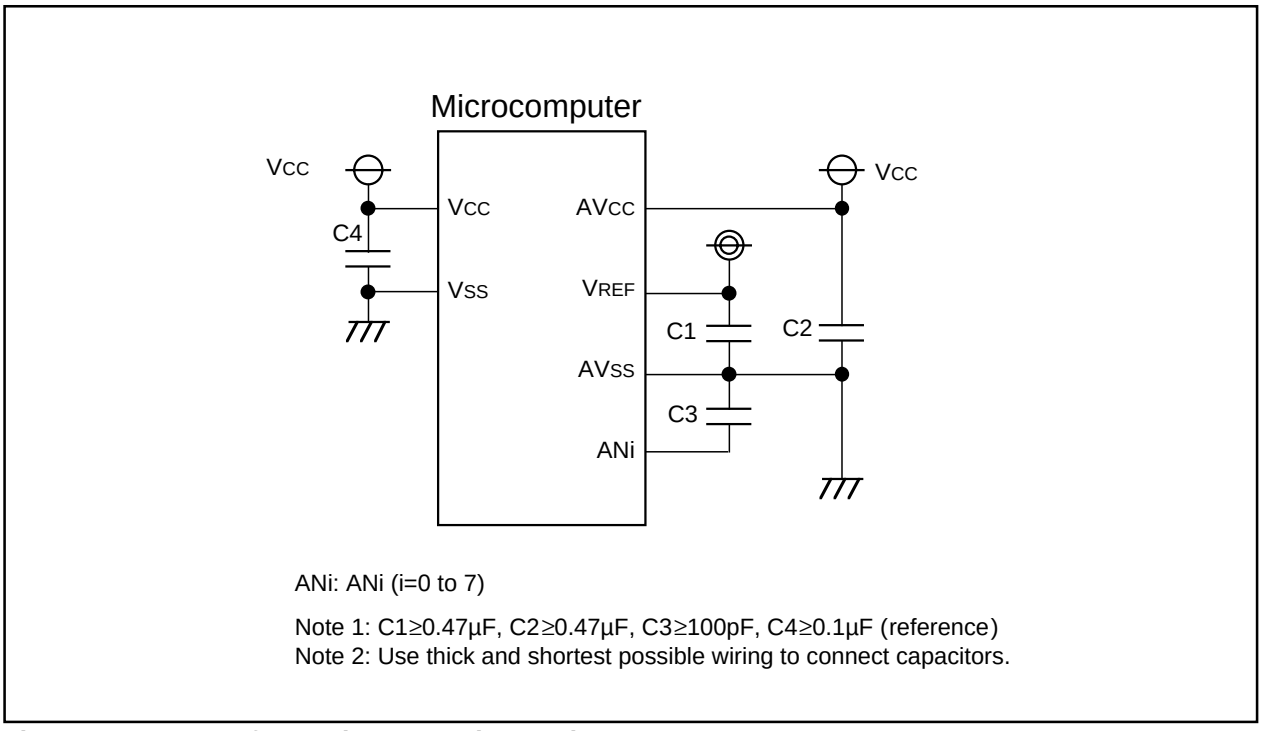
3. To prevent noise-induced device malfunction or latchup, as well as to reduce conversion errors, insert capacitors between the AVCC, VREF, and analog input pins (ANi(i=0 to 7)) each and the AVSS pin. Similarly, insert a capacitor between the VCC pin and the VSS pin. Figure 1.8.1 is an example connection of each pin.

4. Make sure the port direction bits for those pins that are used as analog inputs are set to “0” (input mode). Also, if the ADCON0 register’s TGR bit = 1 (external trigger), make sure the port direction bit for the $\overline{\text{ADTRG}}$ pin is set to “0” (input mode).

5. When using key input interrupts, do not use any of the four AN4 to AN7 pins as analog inputs. (A key input interrupt request is generated when the A-D input voltage goes low.)

6. The ϕAD frequency must be 10 MHz or less. Without sample-and-hold function, limit the ϕAD frequency to 250kHz or more. With the sample and hold function, limit the ϕAD frequency to 1MHz or more.

7. When changing an A-D operation mode, select analog input pin again in the CH2 to CH0 bits of ADCON0 register and the SCAN1 to SCAN0 bits of ADCON1 register.



ANi: ANi (i=0 to 7)

Note 1: C1≥0.47µF, C2≥0.47µF, C3≥100pF, C4≥0.1µF (reference)
Note 2: Use thick and shortest possible wiring to connect capacitors.

Figure 1.8.1. Use of capacitors to reduce noise

8. If the CPU reads the ADi register (i = 0 to 7) at the same time the conversion result is stored in the ADi register after completion of A-D conversion, an incorrect value may be stored in the ADi register. This problem occurs when a divide-by-n clock derived from the main clock or a subclock is selected for CPU clock.
   - When operating in one-shot or single-sweep mode
     Check to see that A-D conversion is completed before reading the target ADi register. (Check the ADIC register's IR bit to see if A-D conversion is completed.)
   - When operating in repeat mode or repeat sweep mode 0 or 1
     Use the main clock for CPU clock directly without dividing it.

9. If A-D conversion is forcibly terminated while in progress by setting the ADCON0 register's ADST bit to "0" (A-D conversion halted), the conversion result of the A-D converter is indeterminate. The contents of ADi registers irrelevant to A-D conversion may also become indeterminate. If while A-D conversion is underway the ADST bit is cleared to "0" in a program, ignore the values of all ADi registers.

1.9 Precautions for Programmable I/O Ports

1. If a low-level signal is applied to the P8$_5$/$\overline{\text{NMI}}$/$\overline{\text{SD}}$ pin when the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin enabled), the P7$_2$ to P7$_5$, P8$_0$ and P8$_1$ pins go to a high-impedance state.

2. The input threshold voltage of pins differs between programmable input/output ports and peripheral functions.
   Therefore, if any pin is shared by a programmable input/output port and a peripheral function and the input level at this pin is outside the range of recommended operating conditions V_{IH} and V_{IL} (neither “high” nor “low”), the input level may be determined differently depending on which side—the programmable input/output port or the peripheral function—is currently selected.

3. When three phase motor control function is enabled, it becomes the following when "L" is input to the P8$_5$ pin. When you don't want to do such a change, without using the P8$_5$ pin as a programable I/O port, input "H" to the P8$_5$ pin before setting PD8$_5$ = "L".

   - When the TB2SC register IVPCR1 bit = “1” (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin enabled), the U/ $\overline{\text{U}}$/ V/ $\overline{\text{V}}$/ W/ $\overline{\text{W}}$ pins go to a high-impedance state.
   - When the TB2SC register IVPCR1 bit = “0” (three-phase output forcible cutoff by input on $\overline{\text{SD}}$ pin disabled), the U/ $\overline{\text{U}}$/ V/ $\overline{\text{V}}$/ W/ $\overline{\text{W}}$ pins go to a normal port.

1.10 Precautions for Flash Memory Version

1.10.1 Precautions for Functions to Inhibit Rewriting Flash Memory Rewrite

ID codes are stored in addresses 0FFFDF16, 0FFFE316, 0FFFEB16, 0FFFEF16, 0FFFF316, 0FFFF716, and 0FFFFB16. If wrong data are written to theses addresses, the flash memory cannot be read or written in standard serial I/O mode.

The ROMCP register is mapped in address 0FFFFF16. If wrong data is written to this address, the flash memory cannot be read or written in parallel I/O mode.

In the flash memory version of microcomputer, these addresses are allocated to the vector addresses (H) of fixed vectors.

1.10.2 Precautions for Stop mode

When shifting to stop mode, the following settings are required:

• Set the FMR01 bit to “0” (CPU rewrite mode disabled) and disable DMA transfers before setting the CM10 bit to “1” (stop mode).

• Execute the JMP.B instruction subsequent to the instruction which sets the CM10 bit to “1” (stop mode)

```
Example program     BSET    0, CM1      ; Stop mode
                    JMP.B   L1
                L1:
                    Program after returning from stop mode
```

1.10.3 Precautions for Wait mode

When shifting to wait mode, set the FMR01 bit to “0” (CPU rewrite mode diabled) before executing the WAIT instruction.

1.10.4 Precautions for Low power dissipation mode, ring oscillator low power dissipation mode

If the CM05 bit is set to “1” (main clock stop), the following commands must not be executed.

• Program

• Block erase

1.10.5 Writing command and data

Write the command code and data at even addresses.

1.10.6 Precautions for Program Command

Write ‘xx4016’ in the first bus cycle and write data to the write address in the second bus cycle, and an auto program operation (data program and verify) will start. Make sure the address value specified in the first bus cycle is the same even address as the write address specified in the second bus cycle.

1.10.7 Operation speed

Before entering CPU rewrite mode (EW0 or EW1 mode), select 10 MHz or less for CPU clock using the CM0 register's CM06 bit and CM1 register's CM17–6 bits. Also, set the PM1 register's PM17 bit to 1 (with wait state).

1.10.8 Instructions inhibited against use

The following instructions cannot be used in EW0 mode because the flash memory's internal data is referenced: UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

1.10.9 Interrupts

EW0 Mode

- Any interrupt which has a vector in the variable vector table can be used providing that its vector is transferred into the RAM area.
- The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used because the FMR0 register and FMR1 register are initialized when one of those interrupts occurs. The jump addresses for those interrupt service routines should be set in the fixed vector table.

  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ or watchdog timer interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.
- The address match interrupt cannot be used because the flash memory's internal data is referenced.

EW1 Mode

- Make sure that any interrupt which has a vector in the variable vector table or address match interrupt will not be accepted during the auto program or auto erase period.
- Avoid using watchdog timer interrupts.
- The $\overline{\text{NMI}}$ interrupt can be used because the FMR0 register and FMR1 register are initialized when this interrupt occurs. The jump address for the interrupt service routine should be set in the fixed vector table.

  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

1.10.10 How to access

To set the FMR01, FMR02, or FMR11 bit to "1", write "0" and then "1" in succession. This is necessary to ensure that no DMA transfers will occur before writing "1" after writing "0". Also when PM24 is "0" (P8$_5$ function($\overline{\text{NMI}}$ disable)) or when PM24 is "1" ($\overline{\text{NMI}}$ function) and $\overline{\text{NMI}}$ pin is "H" level.

1.10.11 Writing in the user ROM area

EW0 Mode

- If the power supply voltage drops while rewriting any block in which the rewrite control program is stored, a problem may occur that the rewrite control program is not correctly rewritten and, consequently, the flash memory becomes unable to be rewritten thereafter. In this case, standard serial I/O or parallel I/O mode should be used.

EW1 Mode

- Avoid rewriting any block in which the rewrite control program is stored.

1.10.12 DMA transfer

In EW1 mode, make sure that no DMA transfers will occur while the FMR0 register's FMR00 bit = 0 (during the auto program or auto erase period).

1.10.13 Regarding Programming/Erasure Times and Execution Time

As the number of programming/erasure times increases, so does the execution time for software commands (Program, and Block Erase). Especially when the number of programming/erasure times exceeds 1,000, the software command execution time is noticeably extended. Therefore, the software command wait time that is set must be greater than the maximum rated value of electrical characteristics.
The software commands are aborted by hardware reset 1, hardware reset 2, $\overline{\text{NMI}}$ interrupt, and watchdog timer interrupt. If a software command is aborted by such reset or interrupt, the block that was in process must be erased before reexecuting the aborted command.

1.10.14 Definition of Programming/Erasure Times

"Number of programs and erasure" refers to the number of erasure per block.
If the number of program and erasure is n (n=100 1,000 10,000) each block can be erased n times.
For example, if a 2K byte block A is erased after writing 1 word data 1024 times, each to a different address, this is countedas one program and erasure. However, data cannot be written to the same adrress more than once without erasing the block. (Rewrite prohibited)

1.10.15 Flash Memory Version Electrical Characteristics 10,000 E/W cycle products (D7, D9, U7, U9)

When Block A or B E/W cycles exceed 100 (D7, D9, U7, U9), select one wait state per block access.
When FMR17 is set to "1", one wait state is inserted per access to Block A or B - regardless of the value of PM17. Wait state insertion during access to all other blocks, as well as to internal RAM, is controlled by PM17 - regardless of the setting of FMR17.
To use the limited number of erasure efficiently, write to unused address within the block instead of rewite. Erase block only after all possible address are used. For example, an 8-word program can be written 128 times before erase becomes necessary.
Maintaining an equal number of erasure between Block A and B will also improve efficiency.
We recommend keeping track of the number of times erasure is used.

1.10.16 Precaution of Boot Mode

The value which isn't fixed is sometimes output in the I/O port until an internal power supply becomes stable when "H" is input to the CNV$_{SS}$ pin and "L" is input to the $\overline{\text{RESET}}$ pin under condition that a internal power supply isn't stable at the time of the power supply injectiopn.
When setting the CNV$_{SS}$ pin to "H", the following settings are required:

(1) Apply an "L" signal to the $\overline{\text{RESET}}$ pin and the CNV$_{SS}$ pin.
(2) The V$_{CC}$ is more than 2.7V, and wait beyond ec 2msec. (Internal power supply stable waiting time)
(3) Apply an "H" signal to the CNV$_{SS}$ pin.
(4) Apply an "H" signal to the $\overline{\text{RESET}}$ pin.

2. Differences Made Depending on Manufactured Time

2.1 RESET Input

Ensure that pin RESET must hold valid-low state during powering-up.

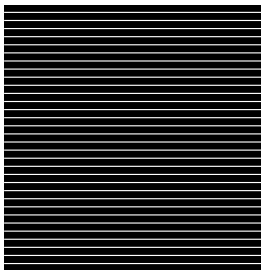
When using a reset IC, use a CMOS type IC. When using an open-drain type reset IC, insert a capacitor between the reset input and Vss and a resistor between the input and Vcc respectively. The R-C time constant of the capacitor and resistor must provide a low state at least 10 times longer than the Vcc rise time.

RENESAS 16-BIT SINGLE-CHIP MICROCOMPUTER
USAGE NOTES REFERENCE BOOK
M16C/26 Group Rev.0.90

Editioned by
Committee of editing of RENESAS Semiconductor Usage Notes Reference Book

---

M16C/26 Group
Usage Notes Reference Book



Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan